

# Image Retrieval with a Bayesian Model of Relevance Feedback

Dorota Glowacka  
Department of Computer  
Science  
University of Helsinki  
glowacka@cs.helsinki.fi

Yee Whye Teh  
Department of Statistics  
University College  
University of Oxford  
y.w.teh@stats.ox.ac.uk

John Shawe-Taylor  
Department of Computer  
Science  
University College London  
jst@cs.ucl.ac.uk

## ABSTRACT

A content-based image retrieval system based on multinomial relevance feedback is proposed. The system relies on an interactive search paradigm where at each round a user is presented with  $k$  images and selects the one closest to their ideal target. Two approaches, one based on the Dirichlet distribution and one based the Beta distribution, are used to model the problem motivating an algorithm that trades exploration and exploitation in presenting the images in each round. Experimental results show that the new approach compares favourably with previous work.

## Categories and Subject Descriptors

H.3.m [Information Search and Retrieval]: Miscellaneous

## General Terms

Image Retrieval

## Keywords

Relevance Feedback; Bayesian Modeling; Dirichlet Distribution; Content-Based Image Retrieval; Exploration/Exploitation

## 1. INTRODUCTION

We consider content-based image (CBIR) retrieval in the case when the user is unable to specify the required content through tags or other image properties. In this type of scenario, the system must extract information from the user through limited feedback. We consider a protocol that operates through a sequence of rounds in each of which a set of images is displayed and the user must indicate which image is closest to their ideal target image(s). Note that we do not always assume that the target image(s) is/are in the database. Instead, we test two hypotheses as to what constitutes the user's target. In our first hypothesis, we assume there is a hypothetical target single image in the user's

mind. Our second hypothesis assumes that there is a set of images, any of which will satisfy the user. Clearly, in both scenarios, the user search concludes in a single image. However, each of the two hypotheses differs in terms of how the user perceives the search procedure. In the former case, the system's assumption is that the user has a precise image in their mind from the beginning of the search session and so the system will try to converge on a particular single image from the first iteration of the search. Unfortunately, this approach might fail if the user is not very familiar with the database they are searching or does not concentrate on finding a specific image and so uses the initial couple of iterations in a more exploratory fashion in order to "familiarise" himself with the image database and better formulate his search target. In the latter hypothesis, the user has a vague notion of the kind of image that is required and wants to explore the current database while at the same time trying to decide what the final target image should be. While the problem of image retrieval with relevance feedback has been studied before (e.g. [3, 6, 8]), we present a novel Bayesian approach that uses latent random variables to model the system's imperfect knowledge about the user's expected response to the images. The proposed approach compares favourably with previous work.

For our first hypothesis, suppose we are given a database  $\mathcal{D}$  of images. For each image  $x_i \in \mathcal{D}$ , we use a latent variable  $\theta_i$  taking values in  $[0, 1]$  to represent the chance that  $x_i$  is the image the user is searching for. Supposing that the user has a single ideal target, the variables have to sum to one:  $\sum_{x_i \in \mathcal{D}} \theta_i = 1$ . Since the system has incomplete knowledge about the user's target image, it uses a Dirichlet distribution over the variables ( $\theta_i$ ) to represent the state of its knowledge. Thus we call our algorithm *Dirichlet Sampling* (DS) [10]. At each iteration, the system samples  $k$  images to present to the user, from which the user selects the one closest to the target. An important aspect of the DS algorithm is the way in which its knowledge of the target is updated given user feedback. We considered two algorithms to do so: variational Bayes and Gibbs sampling.

We compare the performance of the DS algorithm with that of *Beta Experts* (BE), where uniform updates are used to improve the system's knowledge of the target. The assumption of the BE algorithm is that instead of there being a single target image, there is a set of images, any of which will satisfy the user. Thus, we give uniform updates to all the images that are considered to be the user's preferred images. For each image  $x_i \in \mathcal{D}$ , there is a random variable  $b_i$  which models whether image  $x_i$  is a preferred

image. The probability that  $x_i$  is a preferred image is modelled using a number  $p_i \in [0, 1]$ . Since  $p_i$  is unknown, we model it as an unobserved random variable with a beta prior  $\propto p_i^{a-1}(1-p_i)^{b-1}$ . We show in experiments that the simple uniform update performs best. The reason is because both variational Bayes and Gibbs sampling tend to focus on a small set of images (which may or may not contain the target) aggressively, while the uniform update is less aggressive.

An important aspect of both approaches is the incorporation of an explicit exploration-exploitation strategy in the image sampling process, which greatly improves the performance of the algorithms compared to their main competitors that do not employ exploration-exploitation strategies. Our approach is similar to Thompson sampling [5]. In the case of the DS algorithm, a sample is drawn from the Dirichlet distribution which represents the knowledge state of the system, adjusted using a temperature to trade-off exploration and exploitation, and the image with the largest probability (in the sample) of being the target is selected; this is repeated  $k$  times to obtain the  $k$  images presented. We employ a similar approach in the case of the BE algorithm, where we sample  $k$  images from the Beta distribution.

In summary, our system supports the user in finding an image that best matches their query in the manner described below. In each iteration,  $k$  images from the database  $\mathcal{D}$  are presented to the user and the user selects the most relevant image from this set, according to the following protocol: For each iteration  $i = 1, 2, \dots$  of the search:

- Search engine calculates a set of images  $\mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,k} \in \mathcal{D}$  to present to the user.
- If one of the images matches the user’s query, then the search terminates.
- Otherwise, the user chooses one of the images  $\mathbf{x}_i^*$  as most relevant according to a distribution  $D\{\mathbf{x}_i^* = \mathbf{x}_{i,j} \mid \mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,k}; \mathbf{t}\}$ , where  $\mathbf{t}$  denotes the target image. Note that  $\mathbf{t}$  can be either a single image or a representation of the target set of images.

## 2. THE DIRICHLET SEARCH ALGORITHM

In this section, we introduce the main aspects of the DS algorithm. Let  $\mathcal{D}$  be a dataset of  $n$  images  $(\mathbf{x}_i)_{i=1,\dots,n}$ . Let  $M = \{m_1, \dots, m_n\}$  the base measure defined on  $\mathcal{D}$  and  $\alpha$  the number of pseudocounts. Initially, we set  $(m_i)_{i=1,\dots,n} = \frac{1}{n}$  and  $\alpha = 1$ . Let  $\mathbf{x}_j^* \in \{\mathbf{x}_{j,1}, \dots, \mathbf{x}_{j,k}\}$  be the image chosen by the user at iteration  $j$  from among the  $k$  presented images  $\{\mathbf{x}_{j,1}, \dots, \mathbf{x}_{j,k}\}$ . We suppress the index  $j$  to simplify the exposition. In our model, the user only sees  $k$  images at each iteration and so we can only observe user’s preference with respect to these  $k$  images. However, we want to be able to model the user’s preferences with respect to the entire dataset of images. Thus, we view the set of images  $\{\mathbf{x}_1, \dots, \mathbf{x}_k\}$  as partitioning the complete space of images into sets  $\mathcal{X}_1, \dots, \mathcal{X}_k$  with

$$\mathcal{X}_j = \{\mathbf{x} : d(\mathbf{x}_j, \mathbf{x}) < d(\mathbf{x}_{j'}, \mathbf{x}), j' \neq j\}. \quad (1)$$

In order to be able to update the base measures of all images in a given partition at each iteration and be able to approximate the true Dirichlet posterior, we derive the base measure updates using Variational Bayes (VB) [2, 4, 11, 12]. In Section 6.2, we also compare the performance of the DS

algorithm with VB updates and updates obtained through Gibbs sampling.

### 2.1 VB Parameter Updates

Below we illustrate the factorized variational approximation for the Dirichlet Search algorithm. Dirichlet distributions  $Dir(\alpha)$  are probability distributions over multinomial parameter vectors. The distribution is parametrised by a vector  $\alpha = \{\alpha_1, \dots, \alpha_n\}$ , where  $\alpha = (\alpha_1, \dots, \alpha_n) = \alpha_0(m_1, \dots, m_n)$ , where  $(m_1, \dots, m_n)$  has 1-norm 1 and  $\alpha_0 > 1$ . The Dirichlet distribution is conjugate to the Multinomial distribution, which gives us the following generative model:

$$\begin{aligned} \theta | \alpha &\sim Dir(\alpha) \\ \beta_i | \theta &\sim Mult(\theta) \end{aligned}$$

As mentioned earlier, at each iteration, the user is presented with  $k$  images and selects one of them as being “most similar” to the their ideal target image. The selected image is a “proxy” for similar images, which we also consider to be selected by the user. Thus, we want to apply different updates depending on whether a given image was in a partition  $\pi$  selected by the user. Given a set  $\{x_1, \dots, x_k\}$  of observed images, the user selects image  $x_j^*$ , which is the proxy for the partition containing that particular image. We denote the partition containing image  $x_j^*$  as  $\pi_{x_j^*}$ . Our generative model looks as follows:

$$P(\theta | \alpha) \propto \prod_i \theta_i^{\alpha_i - 1} \quad (2)$$

$$P(\beta | \theta) = \prod_i \theta_i^{\beta_i} \quad (3)$$

$$P(x_j^* | \beta, \pi) = \delta \begin{cases} 1 & \text{if } \beta_i \in \pi_{x_j^*} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Given our variables  $\theta$  and  $\beta$ , and the selected image  $x_j^*$ , we wish to obtain  $p(\theta, \beta | x_j^*)$ . For most models, this is intractable and so we consider distribution  $q(\theta, \beta)$ , which can be factorised as:

$$q(\theta, \beta) = q_\theta(\theta) q_\beta(\beta) \quad (5)$$

which gives us the following definition of  $F(q)$ :

$$F(q) = \int q_\theta(\theta) q_\beta(\beta) \log \frac{p(x_j^*, \theta, \beta)}{q_\theta(\theta) q_\beta(\beta)} d\theta d\beta \quad (6)$$

The computation of  $q(\theta)$  proceeds by its maximisation of  $F(q)$ :

$$\log q_\theta(\theta) = \int q_\beta(\beta) \log p(x_j^*, \theta, \beta) d\beta \quad (7)$$

We can rewrite Eq. 7 in terms of an expectation:

$$\log q_\theta(\theta) = \mathbb{E}_{q(\beta)} [\log p(x_j^*, \theta, \beta) d\beta] \quad (8)$$

$$q_\theta(\theta) \propto \exp(\mathbb{E}_{q(\beta)} [\log p(x_j^*, \theta, \beta) d\beta]) \quad (9)$$

Applying the same procedure with respect to  $q(\beta)$  will give us:

$$q_\beta(\beta) \propto \exp(\mathbb{E}_{q(\theta)} [\log p(x_j^*, \theta, \beta) d\beta]) \quad (10)$$

We apply the result (8) to find the expression for the optimal factor  $q_\theta(\theta)$ . We only need to retain the those terms that

have functional dependency on  $\theta$  as all the remaining terms are absorbed into the normalising constant. Thus, we have:

$$\log q_\theta(\theta_i) = \mathbb{E}_{q(\beta_i)}[\log p(\theta_i) + \log p(\beta_i)] + C \quad (11)$$

$$= \mathbb{E}_{q(\beta_i)}[\log(\theta_i^{\alpha_i-1}) + \log(\theta_i^{\beta_i})] + C \quad (12)$$

$$= \mathbb{E}_{q(\beta_i)}[(\alpha_i - 1 + \beta_i) \log \theta_i] + C \quad (13)$$

$$= \alpha_i - 1 + \mathbb{E}[\beta_i \log \theta_i] + C \quad (14)$$

We can identify that

$$q_\theta(\theta_i) = \text{Dir}(\theta|\alpha_i) \quad (15)$$

where

$$\alpha_i^* = \alpha_i + \mathbb{E}[\beta_i] \quad (16)$$

Similarly, we can obtain the expression for the optimal  $q_\beta(\beta)$ :

$$\log q_\beta(\beta_i) = \mathbb{E}_{q(\theta_i)}[\log(\beta_i) + \log(x_j^*)] + C \quad (17)$$

$$= \mathbb{E}_{q(\theta_i)}[\log(\theta_i^{\beta_i}) + \log \delta] + C \quad (18)$$

$$= \mathbb{E}_{q(\theta_i)}[\beta_i \log \theta_i + \log \delta(x_i)] + C \quad (19)$$

$$= \beta_i \mathbb{E}[\log \theta_i] + \log \delta(x_i) + C \quad (20)$$

We can see that

$$q_\beta(\beta_i) = \text{Mult}(\beta_i|\gamma_i) \quad (21)$$

where

$$\gamma_i \propto \begin{cases} 0 & \text{if } x_i \notin \pi_{x_j^*} \\ e^{\mathbb{E}[\log \theta_i]} & \text{otherwise} \end{cases} \quad (22)$$

Note that for Dirichlet distributions [1],

$$e^{\mathbb{E}[\log \theta_i]} \approx \max(0, \alpha_i - .5) \quad (23)$$

so that the update for  $\alpha_i$  is approximately:

$$\alpha_i^* \approx \alpha_i + \begin{cases} \frac{\max(0, \alpha_i - .5)}{\sum_{i: x_i \in \mathcal{X}_j} \max(0, \alpha_i - .5)} & \text{if } x_i \in \mathcal{X}_j \\ 0 & \text{otherwise} \end{cases} \quad (24)$$

In other words, the parameters of images in the chosen partition are incremented, with the total increment equal to 1. Further, images  $x_i$  whose parameter  $\alpha_i$  is already large tend to get a larger share of the increment, while those with small  $\alpha_i$  will not get much increment at all. This effect, where the “rich-get-richer” can easily force the VB algorithm into a situation where the parameters for a small set of images dominates, even though they are not the true image. The resulting image search algorithm will then never converge to the true image since it will always show one of these dominating images.

To address this problem of premature (and wrong) convergence, we propose a simple approach whereby all images in the chosen partition get incremented by an equal amount. The approach is based on the Beta distribution and we call it the Beta Experts algorithm.

### 3. THE BETA EXPERTS ALGORITHM

Let  $\mathcal{D}$  index the set of images. Instead of assuming that the user’s (latent) preference is a single true image, we will assume that there is a set of images, any of which will satisfy the user, i.e. if the user saw one of these images during one of the rounds, the procedure would terminate with success. In other words, for each  $x_i \in \mathcal{D}$ , there is a

binary indicator  $b_i^* \in \{0, 1\}$ , where  $b_i^* = 1$  if image  $x_i$  is one of the user’s preferred images, and  $b_i^* = 0$  otherwise.

For each image  $x_i \in \mathcal{D}$ , let  $b_i$  be a random variable which will model whether image  $x_i$  is a preferred image or not. The value of  $b_i$  is unknown to the system thus modelled as an unobserved random variable.

The probability that  $x_i$  is a preferred image, i.e. that  $b_i = 1$ , is modelled using a number  $p_i \in [0, 1]$ . Since  $p_i$  is unknown, we will also model it as an unobserved random variable, with a beta prior  $\propto p_i^{a-1}(1-p_i)^{b-1}$ .

The system shows the user  $k$  images in each of rounds  $r = 1, \dots, R$ . In round  $r$  the  $k$  images are indexed by  $C_r = \{x_{i_{r1}}, \dots, x_{i_{rk}}\}$ . The user either sees an image she is looking for (a preferred image), in which case the system terminates with success, or the user will choose an image  $k_r$  from among the  $k$  shown that is “closest” to the preferred images in her mind.

We model the user’s choices as follows. At round  $r$  let  $I_{r1}, \dots, I_{rk}$  be a partition of the set of images  $\mathcal{D}$ , such that  $I_{rj}$  contains all images such that  $x_{i_{rj}}$  is the closest image to it among  $x_{i_{r1}}, \dots, x_{i_{rk}}$ . We model the joint probability of choices and preference probabilities as an exponential family harmonium:

$$P(k_1, \dots, k_R, \{p_i\}_{i \in \mathcal{D}}) \propto \prod_{i \in \mathcal{D}} p_i^{a-1}(1-p_i)^{b-1} \prod_{r=1}^R \prod_{i \in I_{rk_r}} p_i \prod_{i \notin I_{rk_r}} (1-p_i) \quad (25)$$

Given the choices of the users up to round  $R$ , the posterior distribution over  $\{p_i\}$  is:

$$P(\{p_i\}_{i \in \mathcal{D}} | k_1, \dots, k_R) \propto \prod_{i \in \mathcal{D}} p_i^{a-1}(1-p_i)^{b-1} \prod_{r=1}^R \prod_{i \in I_{rk_r}} p_i \prod_{i \notin I_{rk_r}} (1-p_i) \quad (26)$$

$$\propto \prod_{i \in \mathcal{D}} p_i^{a+n_i-1}(1-p_i)^{b+R-n_i-1} \quad (27)$$

where  $n_i = \#\{r : i \in I_{rk_r}\}$  is the number of user choices such that  $i$  is in the closest partition. Incidentally, note that the conditional probability that the user will pick  $k_r$  out of the  $k$  images at round  $r$  is:

$$P(k_r | \{p_i\}) \propto \prod_{i \in I_{rk_r}} p_i \prod_{i \notin I_{rk_r}} (1-p_i) \quad (28)$$

$$\propto e^{\sum_{i \in I_{rk_r}} \log p_i / (1-p_i)} \quad (29)$$

is a softmax of the sum of log odds-ratios over images in partition  $I_{rk_r}$ .

In Section 6.2, we compare the performance of the DS algorithm with the VB updates and the BE algorithm.

## 4. BALANCING EXPLORATION VERSUS EXPLOITATION

The final ingredient of the DS and the BE algorithms is how the images presented to the user should be chosen. This involves a trade-off between presenting images that appear promising based on best current estimates of the mean given by the posterior measure (exploitation), and trying areas where our current estimate could be too pessimistic (exploration). The strategy we adopt to solve this

problem is to draw  $k$  samples from the posterior distribution, where each sample corresponds to distribution over all the images, and select the images  $\mathbf{x}_j$ , where  $j = 1, \dots, k$ , that have the highest probability in each of these samples. In case of the DS algorithm, we obtain this by sampling from the Dirichlet distribution. A fast method to sample a random vector from a  $n$ -dimensional Dirichlet distribution with parameters  $m = \{\alpha m_1, \dots, \alpha m_n\}$  is to draw  $n$  independent random samples from the gamma distribution:  $f_i \sim \text{Gamma}(\alpha m_i, 1) = \frac{f_i^{\alpha m_i - 1} e^{-f_i}}{\Gamma(\alpha m_i)}$ , and normalise the resulting vector. Since we are interested only in the maximum, we can omit the normalisation step (Algorithm 1).

---

**Algorithm 1** Image selection for the DS algorithm

---

```

for  $j = 1, \dots, k$  do
  for  $i = 1, \dots, n$  do
     $f_i \leftarrow \text{randg}(\alpha m_i)$ 
  end for
   $[\text{value}_j, \text{index}_j] \leftarrow \max(f)$ ;  $\text{images}_j \leftarrow \text{index}_j$ 
end for
Return: array images with indices of selected images

```

---

In case of the BE algorithm, we draw  $n$  independent samples from the beta distribution and select the maximum (Algorithm 2).

---

**Algorithm 2** Image selection for the BE algorithm

---

```

for  $j = 1, \dots, k$  do
  for  $i = 1, \dots, n$  do
     $r_{a_i} \leftarrow \text{randg}(a_i)$ 
     $r_{b_i} \leftarrow \text{randg}(b_i)$ 
     $f_i = \frac{r_{a_i}}{r_{a_i} + r_{b_i}}$ 
  end for
   $[\text{value}_j, \text{index}_j] \leftarrow \max(f)$ ;  $\text{images}_j \leftarrow \text{index}_j$ 
end for
Return: array images with indices of selected images

```

---

## 5. THE USER MODEL

We assume that the choice of one of the presented images is a random process, where more relevant images are more likely to be chosen. This models some source of noise in the user's selection process. In our simulation experiments reported in the next section, we will rely on the user model proposed by [3]. The prediction of this particular user model has been shown to be compatible with the actual user behaviour [9]. In Section 6.4, we also compare the performance of the algorithms in real-life and in simulations. Following [3], we assume a similarity measure  $S(\mathbf{x}_1, \mathbf{x}_2)$  between images  $\mathbf{x}_1, \mathbf{x}_2$ , which also measures the relevance of an image  $\mathbf{x}$  compared to an ideal target image  $\mathbf{t}$  by  $S(\mathbf{x}, \mathbf{t})$ . We assume that in the case of the DS algorithm,  $\mathbf{t}$  is the ideal target image, while in the case of the BE algorithm,  $\mathbf{t}$  is the centre of the partition containing images preferred by the user. Let  $0 \leq \lambda \leq 1$  be the uniform noise in the user's choice. The probability of choosing image  $\mathbf{x}_{i,j}$  is given by:

$$D\{\mathbf{x}_i^* = \mathbf{x}_{i,j} \mid \mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,k}; \mathbf{t}\} = (1 - \lambda) \frac{S(\mathbf{x}_{i,j}, \mathbf{t})}{\sum_{j=1}^k S(\mathbf{x}_{i,j}, \mathbf{t})} + \frac{\lambda}{k}. \quad (30)$$

**Table 1: Comparison of the performance of the DS algorithm with VB updates (VB) and the Beta Experts (BE) algorithm**

	$k = 2$		$k = 5$		$k = 10$	
Target Size	BE	VB	BE	VB	BE	VB
1	69	450	26	198	17	86
5	47	431	19	89	14	38
10	41	393	17	60	11	22

Assuming a distance function  $d(\cdot, \cdot)$ , a possible choice for the similarity measure  $S(\cdot, \cdot)$  is

$$S(\mathbf{x}, \mathbf{t}) = d(\mathbf{x}, \mathbf{t})^{-a} \quad (31)$$

with parameter  $a > 0$ . This particular measure decreases polynomially with increasing distance. The parameter  $a > 0$  indicates the user "sharpness". Large  $a$  implies that the user favours images closer to the target image. With the polynomial similarity measure, the user's response depends on the relative size of the image distances to the ideal target image.

We use the Euclidean norm

$$d(\mathbf{x}, \mathbf{t}) = \|\mathbf{x} - \mathbf{t}\| \quad (32)$$

as the distance measure between image  $\mathbf{x}$  and the target image  $\mathbf{t}$ . In all the experiments reported below, the values of  $a$  and  $\lambda$  in the user model were kept constant at 2 and 0.1, respectively (these were the optimal values based on [9]).

## 6. EXPERIMENTS

We tested the performance of the DS and the BE algorithms in simulations and in real-life experiments. The aim of the simulation experiments was to assess which one of the proposed models achieves better results, while the aim of the real-life experiments was to confirm the compatibility of the user model with the real user behaviour. In all the experiments, we used a subset of the Tiny Images Dataset [13] with 758 categories and 37900 images. We used the Basic Image Features (BIFs) technique [7] to extract the image features. During the experiments, we measured two things: (1) the average number of iterations required by each method to find the target image; and (2) the average distance of the  $k$  images presented at each iteration from the target image(s).

### 6.1 Simulation experiments

All the reported results from the simulation experiments are averaged over 1000 searches for randomly selected target images from the dataset. In all the experiments reported in this section, we employed the user model described in Section 5.

In the first set of experiments, we compared the performance of the DS algorithm with VB updates with the BE algorithm. Table 1 shows the average number of iterations to find the target image with both algorithms as the value of  $k$  increases. We also measured the number of iterations required by each algorithm to get to the target that consists only of the ideal target image (target size = 1) or the target image plus a number of images close to it. As the experimental results show, the BE algorithm significantly outperforms the DS algorithm with the VB updates. A possible explanation might be the value of the updates, which are smaller in case of VB updates. In case of the BE algorithm, the

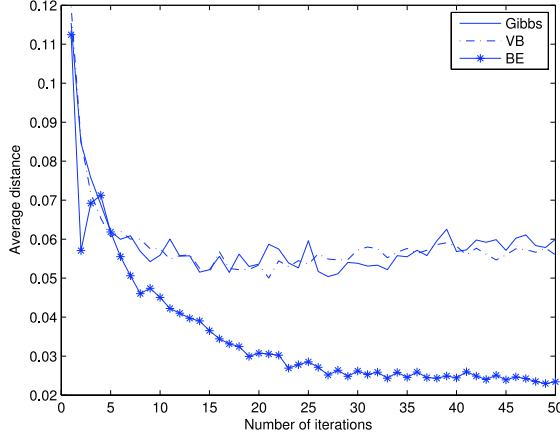


Figure 1: Convergence of the DS algorithm with a Gibbs sampler and VB updates, and the Beta Experts (BE) algorithm.

weights of more promising images are increased by a uniform value of 1, which allows the algorithm to sample more relevant images early on in the search. The other possibility is that VB zooms in on a particular image too quickly and through future updates cannot easily recover from the initial “wrong” choices by the user.

## 6.2 Gibbs Sampling and the Dirichlet Search Algorithm

As the experimental results reported in the previous section show, the DS algorithm with VB updates performs much worse than the BE algorithm. VB provides only an approximate of the “true” posterior. In the next set of experiments, we apply Gibbs sampling to obtain a better approximation of the posterior. Gibbs sampling is applicable when the joint distribution is not known explicitly or is difficult to sample from directly, but the conditional distribution of each variable is known and is easy to sample from. Gibbs sampling generates an instance from the distribution of each variable in turn, conditional on the current values of the other variables. The sequence of samples constitutes a Markov chain, and the stationary distribution of that Markov chain is the sought – after joint distribution that we try to approximate. Due to the Markovian properties, the approximation of the joint distribution derived through Gibbs sampling is usually more accurate than that provided by VB.

The Gibbs sampler that we used in our experiments is summarised in Algorithm 3 below.

Figure 1 shows the average distance from the target of 10 images presented to the user over the first 50 iterations of the algorithm. The results are averaged over 1000 runs of the search algorithm with a random target image selected in each run. As the results show, incorporating Gibbs sampling into the search procedure does not improve the performance of the DS algorithm, indicating that the poor performance is not as a result of the approximate inference but of a poor model, i.e. the correct Bayesian model is inferior to the Beta Experts model, indicating that the assumption that

---

### Algorithm 3 Gibbs sampler for the DS algorithm

---

```

input: base measures  $m_{1,\dots,n} = \frac{1}{n}$ ;
matrix  $C$   $z \times k$  with  $k$  images presented at iterations
 $1, \dots, z$ 
for  $j = 1, \dots, k$  do
  for  $i = 1, \dots, 100$  do
    for  $l = 1, \dots, z$  do
       $s = \sum_{h \in O_l} m_h^{i-1}$ , where  $O_l$  is partition containing
      image  $C_{y,l}$ 
       $r_h = \frac{m_h}{s}$   $h \in O_l$ 
       $v_l \sim \text{Mult}((r_h)_{h \in O_l})$ 
    end for
     $b_h = m_h^{i-1} + \sum_{l=1}^z 1(v_l = h)$ 
     $m_h^s \sim \text{Dir}(b_h)$ 
  end for
   $g_j = \text{Gamma}(m^s)$ 
  [value, index] = max( $g_j$ )
end for
return: array images with indices of selected images

```

---

users have a set of images rather than a single target image is a better reflection of the users’ behaviour.

## 6.3 Related image retrieval algorithms and experimental results

In the last set of simulation experiments, we compare the BE algorithm against two alternatives: the search algorithm proposed in [3] (we call it the AL algorithm) and *PicHunter* [6]. We used the same experimental setting as in the experiments described above. We selected these two algorithms as our benchmark due to the fact that, similarly to our approach, they combine a relevance feedback model with an exploration–exploitation trade-off.

The AL algorithm proposes a weighting scheme that demotes all apparently less relevant images by a constant discount factor  $0 \leq \beta < 1$ . Initially, all the weights  $w_i = 1$ . After each iteration, all the images close to the one selected by the user stay unchanged, while all the remaining images are demoted by  $\beta$ . We set the value of  $\beta = 0.5$  as this was the value that gave the best experimental results.

The *PicHunter* image retrieval system uses Bayes’ rule to predict the user’s target image. The system maintains a set of probabilities for every image. Initially, all the probabilities  $p_i = \frac{1}{n}$ . During each iteration, the search engine displays a set of images and the user selects an image. The probabilities are updated as in  $p_i = p_i * G(d(x_i, s_m))$ , where  $d(x_i, s_m) = \|x_i - s_m\|$  is the distance between image  $x_i$  and image  $s_m$  selected by the user in iteration  $m$ , and  $G$  is defined as

$$G = \frac{\exp(-d(x_i, s_m)/\sigma)}{\sum_{j=1}^n \exp(-d(x_j, s_m)/\sigma)} \quad (33)$$

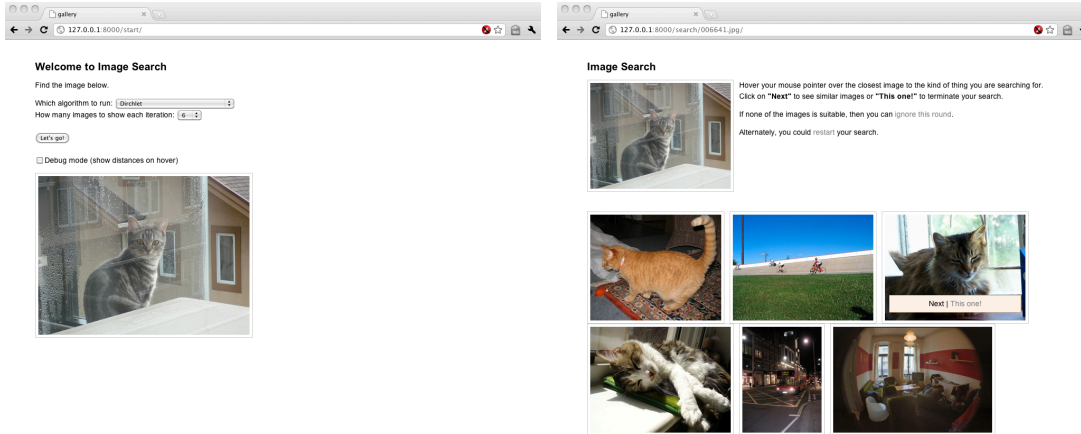
In all the experiments reported here  $\sigma = 0.3$ . After each iteration,  $k$  images with the highest  $p$  are displayed.

We calculated the average number of iterations to terminate the search as the value of  $k$  increases (Table 2). We tested the scaling properties of the algorithm by increasing the size of the image target set, while keeping the size of the image database constant, i.e. the search terminates when one of the 5 or 10 images closest to the target is presented.

The BE algorithm significantly outperforms the AL algorithm and *PicHunter*. The disappointing performance of

**Table 2: Comparison of the AL algorithm, the Beta Experts (BE) algorithm and *PicHunter* (PH)**

	<b>k = 2</b>			<b>k = 5</b>			<b>k = 10</b>		
Target Size	AL	BE	PH	AL	BE	PH	AL	BE	PH
1	227	69	482	117	26	422	92	17	374
5	167	47	461	81	19	390	59	14	328
10	139	41	454	64	17	370	48	11	308



**Figure 2: User interface of the image search prototype system.**

the AL algorithm might be attributed to downgrading the weights by a constant value  $\beta$ . If during the initial iterations the user selects an image far away from the target, the weights of the images close to the target will be demoted early on in the search thus making them less likely to be sampled in the future. *PicHunter* does not really have an exploration stage – rather than sampling images, at each iteration only the  $k$  images with the highest probabilities are displayed. If initially an image far away from the target is selected, then there is little hope for the algorithm to “recover” from the initial bad probability updates as the search progresses.

## 6.4 Real-life Experiments

In order to assess the compatibility between our user model and the real-life user behaviour, we tested the BE algorithm on real users using the same Tiny Images subset as in our simulation experiments. For this purpose, we built a prototype search engine based on the BE algorithm. The system uses a simple user interface designed to search for target images with minimum training. The user interface is shown in Figure 2. At the start of the session, the user is shown the target image that he is expected to search for. The user presses the start button and is taken to the next page where he is presented with  $k$  images. In the example shown in Figure 2, six images are displayed at each iteration. The target image is always present in the left top corner of the display to avoid possible interference from memory problems. The user selects the image that is most similar to the target image by clicking “next” on that particular image. The process is repeated until the desired image is found, at which point the user clicks “this one!” on the selected image.

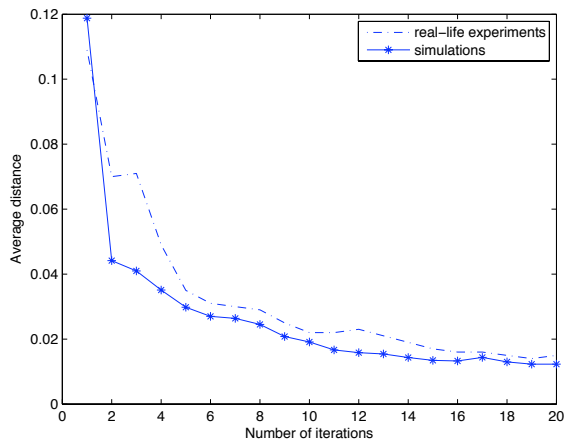
The system was tested on 15 users who performed 4 searches each using the Beta Experts algorithm. At each iteration, 10

images were displayed. The users were instructed to terminate the search when they found the target image or after 50 iterations of the search algorithm. If after 50 iterations, the target still has not been found, the user was asked to select the image most similar to the target. The average number of iterations that was required to find the target image was 20. We expect the results to improve if a larger number of images are displayed at each iteration.

In spite of the fact that the target image was always on display during the search process, some of the users continued with the search in spite of the fact that the image they are looking for had already been presented at an earlier iteration, or terminated the search when presented with an image similar to the target image. This observation also supports our hypothesis that users tend to approach the image retrieval task in a more exploratory manner and do not set on a specific target from the very onset of the search (even when explicitly presented with a specific target image). For this reason, at each iteration we calculated the average distance of the currently displayed images from the target image. Our expectation was that at each iteration the average distance would be getting smaller until eventually it flattens out. In order to assess the compatibility of our user model with real-life performance of the system, we plotted the average distance of the displayed images from the target image for the simulations as well as real-life experiments. As Figure 3 shows, in both cases the algorithm displays a similar convergence patterns.

## 7. CONCLUSIONS

We have presented a new approach to content-based image retrieval based on multinomial relevance feedback. We model the knowledge of the system using a Dirichlet pro-



**Figure 3: Comparison of the convergence of the Beta Experts algorithm in real-life experiments and in simulations.**

cess as well as Beta Experts algorithm. Both models suggests an algorithm for generating images for presentation that trades exploration and exploitation. We show that the BE approach that assumes that the user has a set of images as their target is a better reflection of user behaviour than the assumption that the user has a single ideal target in their mind. Furthermore, the experiments confirm that the new approach outperforms earlier work using more heuristic strategies.

## References

- [1] A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh. On smoothing and inference for topic models. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*, 2009.
- [2] H. Attias. A variational bayesian framework for graphical models. In *Proceedings of Neural Information Processing Systems*, 2000.
- [3] P. Auer and A. Leung. Relevance feedback models for content-based image retrieval. In *Multimedia Analysis, Processing and Communications*. Springer, 2009.
- [4] M. J. Beal. Variational algorithms for approximate bayesian inference. PhD Thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.
- [5] O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *Proceedings of Neural Information Processing Systems*, volume 25, 2011.
- [6] I. Cox, M. Miller, T. Minka, T. Papathomas, and P. Yianilos. The bayesian image retrieval system, pichunter: theory, implementation, and psychophysical experiments. *IEEE Transactions on Image Processing*, 9:20 – 37, 2000.
- [7] M. Crosier and L. Griffin. Using basic image features for texture classification. *International Journal of Computer Vision*, 88(3):447 – 460, 2010.
- [8] R. Datta, D. Joshi, J. Li, and J. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 2008.
- [9] D. Glowacka, A. Medlar, and J. Shawe-Taylor. Sifting through images with multinomial relevance feedback. In *NIPS Workshop Beyond Classification: Machine Learning for Next Generation Computer Vision Challenges*, 2010.
- [10] D. Glowacka and J. Shawe-Taylor. Content-based image retrieval with multinomial relevance feedback. In *JMLR Workshop and Conference Proceedings: 2nd Asian Conference on Machine Learning, Tokyo, 2010*, volume 13, pages 111–125, 2010.
- [11] T. S. Jaakkola. Tutorial on variational approximation methods. In M. Oppor and D. Saad, editors, *Advances in Mean Field Methods*, pages 129 – 159. MIT Press, 2001.
- [12] M. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. In M. Jordan, editor, *Learning in Graphical Models*, pages 105 – 162. MIT Press, 1999.
- [13] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: a large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958 – 1970, 2008.